物联网轻量级认证加密算法 ASCON 的软硬件协同设计

汪静[1]，何乐生[1,2]，李忠红[1]，李路迟[1]，杨航[1]
（1. 云南大学信息学院，云南 昆明 650091；2. 云南省高校物联网技术及应用重点实验室，云南 昆明 650091）

摘　要：ASCON 是 2021 年在 NIST 轻量级认证加密征集方案中最有希望成为国际标准的算法，该算法旨在物联网资源受限环境中获得最佳性能，在公开文献中还未见基于该算法的硬件 IP 核实现。提出了一种 ASCON 的软硬件协同实现方法，该方法通过 *S* 盒优化、先验计算和先进的流水线设计等硬件手段提升了 ASCON 在物联网安全认证应用中的速度，同时降低了内存占用。作为对比，在常见的物联网嵌入式处理器平台上软件移植了 ASCON，结果显示所述方法的速度提升了 7.9 倍以上，而存储器的占用则降低了至少 90%。所述方法可用于物联网安全专用集成电路或片上系统（SoC, system on a chip）的设计和实现。

关键词：物联网；ASCON；软硬件协同设计；硬件 IP 核；FPSoC
中图分类号：TN918.4
文献标志码：A
doi: 10.11959/j.issn.2096−3750.2022.00309

Software and hardware co-design of lightweight authenticated ciphers ASCON for the internet of things

WANG Jing[1], HE Lesheng[1,2], LI Zhonghong[1], LI Luchi[1], YANG Hang[1]
1. College of Information, Yunnan University, Kunming 650091, China
2. University Key Laboratory of Internet of Things Technology and Application of Yunnan Province, Kunming 650504, China

Abstract: ASCON was the most promising algorithm to become an international standard in the 2021 NIST lightweight authenticated encryption call for proposals. The algorithm was designed to achieve the best performance in IoT resource-constrained environments, and there was no hardware IP core implementation based on this algorithm in the open literature. A software-hardware collaborative implementation method of ASCON was proposed, which improved the speed and reduced the memory footprint of ASCON in IoT security authentication applications through hardware means such as *S*-box optimization, prior calculation and advanced pipeline design. As a comparison, ASCON has been transplanted on the common IoT embedded processor platform. The results showed that the described method was more than 7.9 times faster, while the memory footprint was reduced by at least 90%. The schemes can be used for the design and implementation of IoT security application-specific integrated circuits or SoCs.

Key words: IoT, ASCON, software and hardware co-design, hardware IP core, FPSoC

0　引言

研究人员尝试用低计算量的密码算法满足资源受限环境下物联网的安全[1-2]。密码算法是物联网数据安全的最佳选择[3]，提供集成机密性和真实性的信息安全是物联网安全的主要挑战[4]。关联数据认证加密可以同时提供机密性、完整性和真实性，是一种兼顾性能和资源的解决方案[5-6]。ASCON 在 2018 年被评为认证加密竞赛（CAESAR, competition for authenticated encryption: security, applicability, and robustness）最终产品组合中轻量级认证加密的主要选择[7-8]，在 2021 年被美国国家标准和技术研

收稿日期：2022−06−21；**修回日期：**2022−11−07
通信作者：何乐生，he_lesheng@263.net
基金项目：国家自然科学基金资助项目（No.U1631121）
Foundation Item: The National Natural Science Foundation of China (No.U1631121)

究院（NIST, National Institute of Standards and Technology）确定为受限设备下轻量级认证加密算法的十种解决办法之一[9-10]。

ASCON 是基于海绵结构的代换–置换网络（SPN, substitution–permutation network）密码算法[8]，采用海绵体结构可以有效防止自适应的密码泄露[11]，而 SPN 保证了算法能以较少的资源实现数据快速扩散已达到良好的密码学性能[12]。该算法主要包括两种变体 ASCON_128 和 ASCON_128A。ASCON 加密过程如图 1 所示，ASCON 解密过程如图 2 所示，ASCON 包括初始化、关联数据处理、明文/密文处理、终结化 4 个部分[8]，算法基本模块是 320 位输入输出的排列函数（permutation）、基本数据处理单元是轮函数（round），初始数据状态由初始向量 IV、密钥 K、公共消息 N 构成，a 和 b 的数值表明排列函数调用的轮函数次数。ASCON_128 和 ASCON_128A 的区别分为两个部分：b 的数值，b=6 时是 ASCON_128 模式，b=8 时是 ASCON_128A 模式；处理的数据块大小，前者每次处理 64 位，后者每次处理 128 位。ASCON 的数据处理状态大小是 320 位的，每经过算法流程的一个步骤，状态更新一次，经过初始化和关联数据的处理之后，将 64 位明文和 320 位状态数据的低 64 位进行异或即可得到 64 位密文，然后再对状态排列更新，继续对下一段明文进行加密。认证加密算法之所以能够对传输过程进行认证，就是因为认证加密算法在输出密文时会附带一个标签，在解密过程中，首先对标签进行验证，在同样的初始向量下如果标签一致那么证明密文在传输过程中没有被更改，具有真实性、完整性和机密性。在 ASCON 中，首先通过初始化和关联数据处理将初始化向量信息和关联数据信息包含进 320 位状态数据，在加强了算法的健壮性的同时，又结合终结化模块的排列和异或操作输出 128 位标签用于实现认证。

在目前 NIST 对轻量级认证加密算法的基准测试中，要么采用纯软件的测试方式，要么采用纯硬件的测试方式[10]，没有考虑到算法作为片上系统功能块[7]的可能性。现场可编程片上系统（FPSoC, field-programmable system-on-chip）是当前片上系统的前沿技术[13]，它由处理系统（PS, processing system）端和可编程逻辑（PL, programmable logic）端[14]两个部分组成。AXI4 总线用于 PL 端与 PS 端的数据通信[15]，通过 AXI4 总线实现 ARM 核（PS）和 FPGA（PL）的协同工作，既提供了 ARM 的灵活性和可拓展性，又通过 FPGA 的保证了工作的高效性，同时软硬件协同工作在一定程度上降低了系统复杂度及技术实现难度[16-17]，提高了系统工作可靠性。FPSoC 具有高性能、低功耗、易拓展的特点，

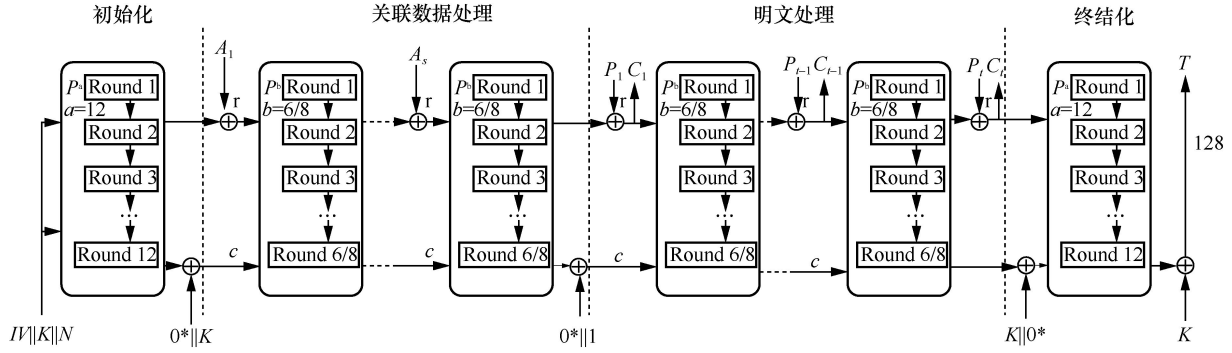


图 1　ASCON 加密过程

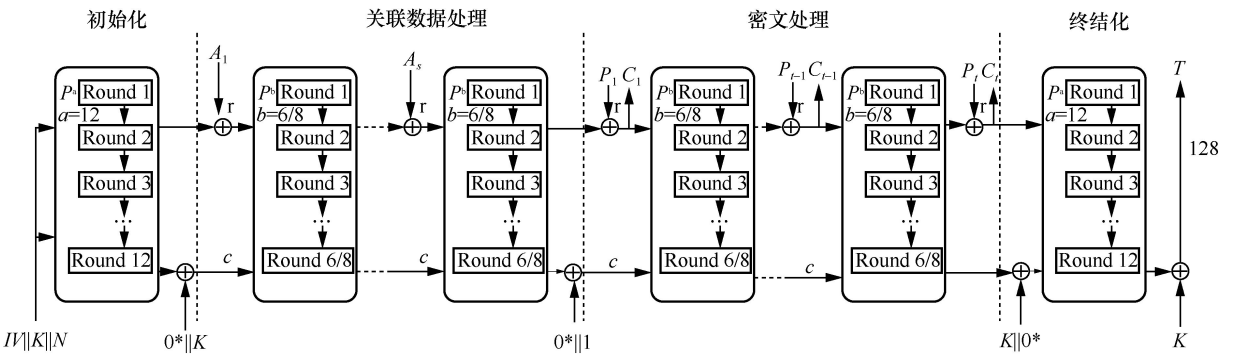


图 2　ASCON 解密过程

是物联网更智能的节点选择[18-22]。文本选用 XILINX 下 ZYNQ-7000 系列的 7020 作为软硬件设计的 FPSoC 硬件平台。设计本身具有可移植性，可在 Altera Cyclone 系列或者其他专用集成电路（ASIC, application specific integrated circuit）上使用。

在对 ASCON 进行软硬件协同工作设计的同时，以 NIST 基准三[23]的性能指标与测试向量对系统进行性能验证，分别在横向和纵向上将基准三测试结果、ARM 核实现、软硬件协同设计进行对比分析。在基准三中，明文和关联数据的测量向量分别从 0 按每次递增 1 byte 的规律增加到 32 byte，以关联数据为内循环，以明文为外循环，一共可产生 33 × 33=1 089 次的测试向量。由于 ARM 核的物联网嵌入式设备大多以 32 位为基本数据处理单元[24-25]，故本文在进行软硬件协同设计性能验证时，在 ZYNQ 的 ARM 上选择以 ASCON 两种变体的 4 种 32 位软件优化版本实现为对比，具体为：32 位速度优化（bi32）、32 位汇编优化（bi32_arm）、32 位寄存器优化（bi32_lowreg）、32 位大小优化（bi32_size）。

1　自定义硬件 IP 核设计

软硬件协同设计分为软件和硬件两个部分[26]，软件部分是配合硬件 IP 核给入测试向量以验证 IP 核设计的正确性。故协同设计的工作重点在硬件的 IP 核设计上，软件设计部分主要在 NIST 给出的测试文件上进行修改，使其能够在 FPSOC 上实现软硬件的协同工作。

自定义 IP 系统架构如图 3 所示。一个完整的 ASCON 加解密功能应该包括初始化模块，关联数据处理模块、加密模块、解密模块、终结化模块，在本文的优化 IP 核设计中运用先验计算省略了初始化模块，且通过模块复用仅用 4 个模块实现了加解密功能，节约了大量硬件资源。时序设计上，采取流水线设计将 ASCON 的加解密功能进行逻辑分割，最终以每 6 个时钟为周期完成一次加密或解密操作，提高了 FPGA 部分的处理速度；同时对不同时序的模块输入信号进行时序对齐，避免了逻辑处理混乱。

1.1　自定义 IP 核的优先策略

AXI 总线包括 AXI4、AXI4_Lite、AXI4_Stream[27]。AXI4-Lite 是一种轻量级的地址映射总线[27]，一个地址传输 4 byte 数据，占用资源是 3 种总线里面最少的。考虑物联网资源受限环境，本文选择 AXI4_Lite 总线。

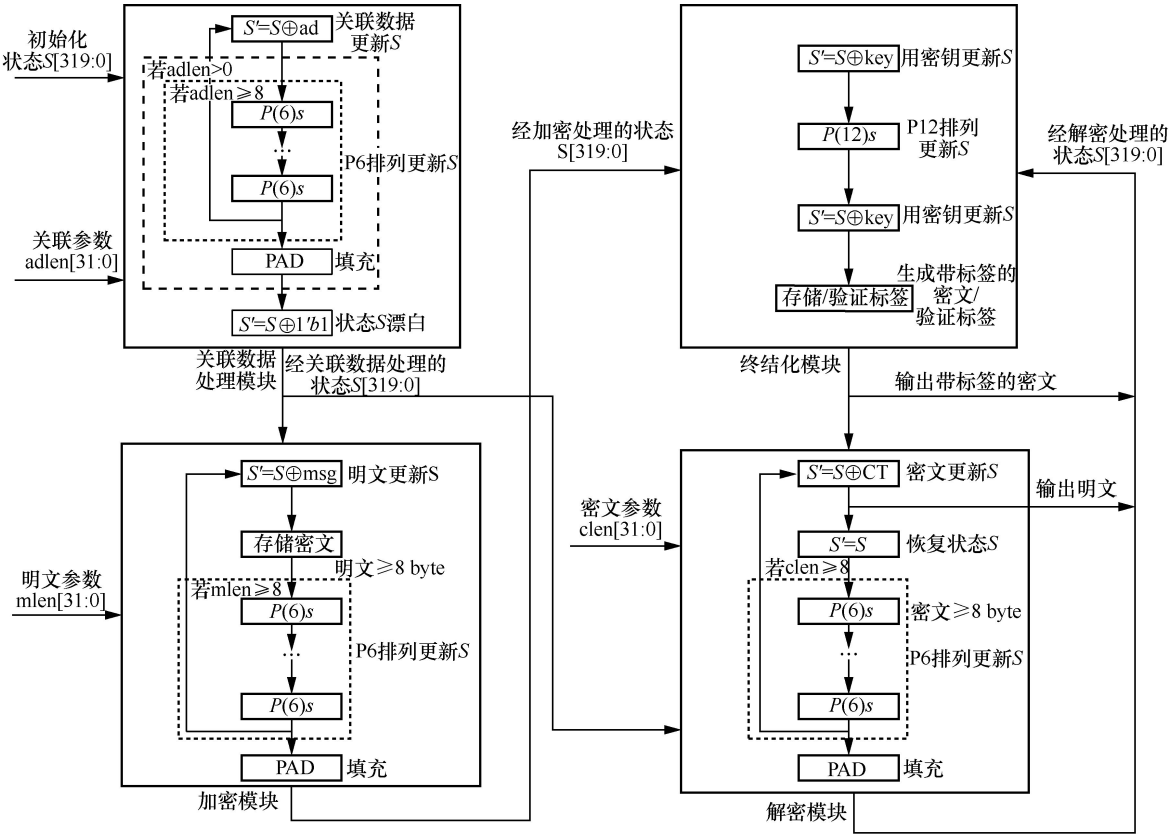


图 3　自定义 IP 系统架构

由于 AXI4_Lite 总线一次最大只能传输 32 位数据，根据算法原理，ASCON 的基本数据处理模块是排列函数。按照传统办法，以核心处理单元排列函数为自定义 IP 核设计，进行 IP 核加速，排列函数的 PS 与 PL 端的数据搬运如图 4 所示，需要进行 2 次 P12 和 9 次 P6 的 IP 核调用，共计进行 22 次 320 位数据传输，即进行了 $22\times(320\div10)=220$ 次数据搬运，极大地增加了 PS 与 PL 之间的数据交互次数，降低了系统整体效率[28-29]。所以这里的 IP 核选择应以减少数据搬运为原则，本文将整个 ASCON 算法作为自定义 IP 核设计的对象，在 PL 和 PS 端只进行必要的参数传输，包括明文参数、关联数据参数、密文参数，对于本文后续的验证只需要传入相关参数即可完成认证加解密的过程，共计传入 192 位数据，搬运数据次数为 $1\times(192\div32)=6$ 次，相比排列函数的 IP 核在数据搬运上节省了 97%左右的时间。

1.2　两种基本函数实现

在 ASCON 结构中，包含两种基本数据处理结构，分别是最小数据处理单元——轮函数和最小数据处理模块——排列函数。排列函数包括 P6.V 和 P8.V 两个部分，它们是在轮函数的基础上进行串行迭代[30]实现的，通过重复调用轮函数，将上次轮函数的输出给到下次轮函数的输入，直到完成对整个数据的排列处理。虽然采用串行迭代优化会削减算法运行速度，但通过模块复用也减少了对 FPGA 硬件资源的消耗。

在最小数据处理单元上，总共有 3 个步骤，分别是轮常量加层、S 盒非线性映射层、线性扩散层[31]，最小数据处理单元——轮函数如图 5 所示。其中轮

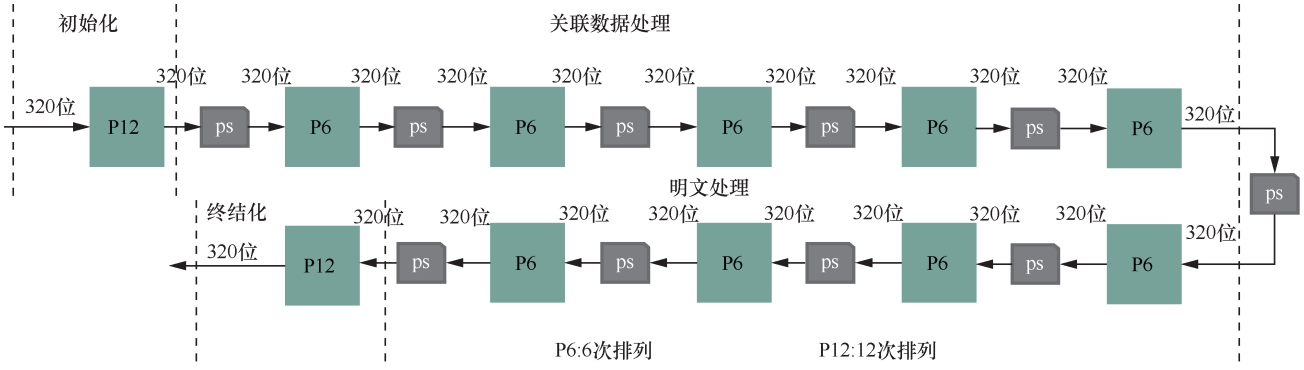


图 4　排列函数的 PS 与 PL 端的数据搬运

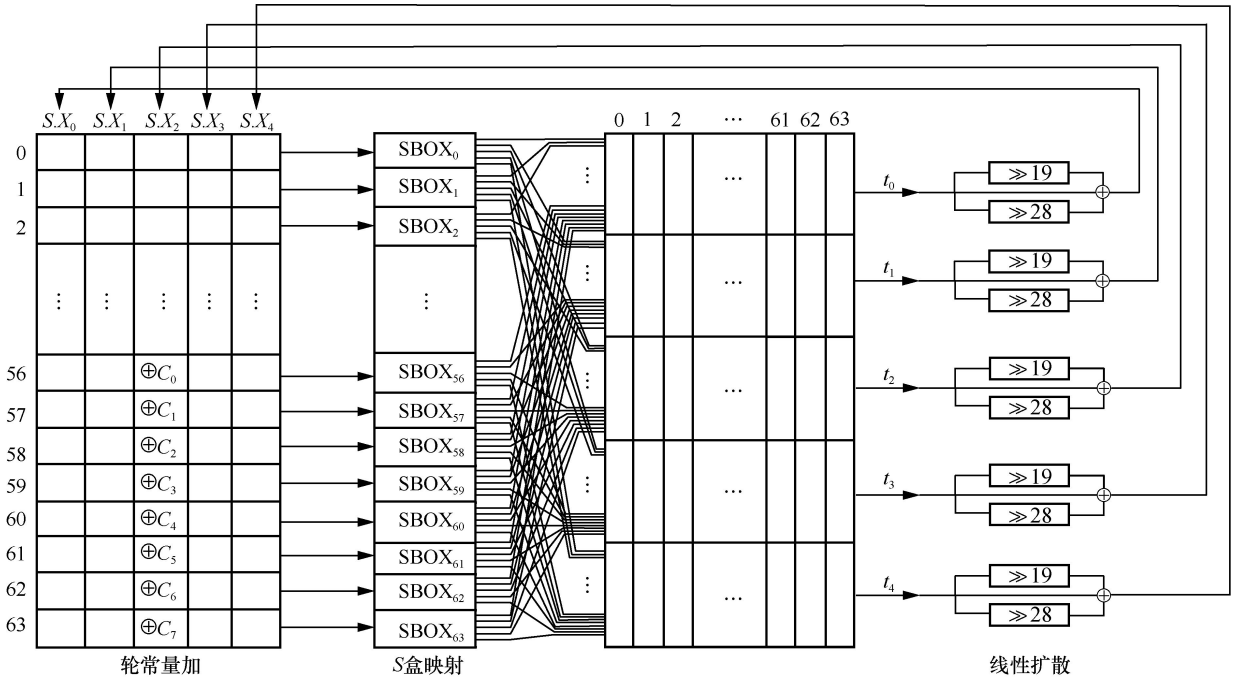


图 5　最小数据处理单元——轮函数

常量加和线性扩散层采用简单的异或和移位逻辑实现[8]，不对其进行时钟分配，减少对 FPGA 功耗的消耗。原 S 盒如图 6 所示，采用是逻辑运算实现，这里本文基于 FPGA 具有丰富的查找表资源和 ASCON 算法设计原理，采用 *S* 盒优化实现，将 5 位 64 个 *S* 盒并行展开，在最小数据处理单元内部尽量提高数据处理速度，*S* 盒优化的查找表见表 1。

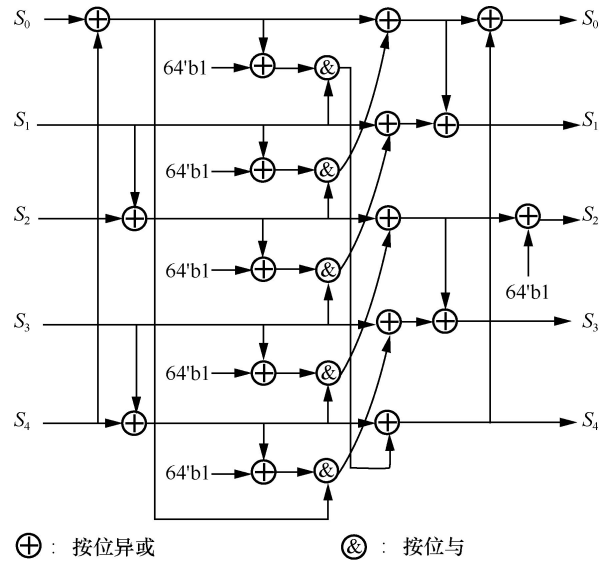


图 6　原 *S* 盒设计

1.3　先验计算优化设计

ASCON 对数据进行处理的开始阶段是初始化。根据前面的 PL 端与 PS 端缓存策略，本文在初始化阶段进行的是一些明文、关联数据参数和 ASCON 配置参数的处理，这些数据一旦确定便具有不变性，所以可以根据这个算法特性直接对这些固定的参数进行 P12 排列函数的先验计算优化设计，得出结果直接放入 FPGA 中，进入到下一轮关联数据的处理。

物联网环境下，关联数据是信息，但不需要额外的保密，不能对其进行随意更改[32]。即关联数据可以公开，但第三方不能更改，如嵌入式设备的唯一 ID。在一个安全的物联网系统中，加/解密双方都可以预先知道关联数据。故将关联数据直接固化在硬件内部，既能做到防止第三方窃取，又能节约硬件资源。关联数据处理需要根据待处理的关联数据的长度进行不同次数的排列函数调用，在已知关联数据的情况下，可以预先计算出关联数据进行排列处理的结果，将结果直接给出，减少不必要的排列调用和时序消耗。

1.4　时序优化设计

时序优化设计部分主要运用了流水线技术[33-34]和时序打拍技术。按照模块功能可分为关联数据处理时序优化设计、加密模块时序优化设计、解密模块时序优化设计。

系统时序上，将排列函数看作一个整体，在一个时钟周期内完成一次排列函数处理，以面积换速度，提高了 ASCON 算法运行速度。ASCON 作为一个完整的算法，包含很多相互关联的模块，FPGA 无法在一个时钟周期处理完所有的数据。因此，有必要采取流水线设计将相互关联的多个模块分割开，在各个部分插入寄存器，将结果缓存起来，等待下一个时钟上升沿再对数据进行处理，对数据逐步加工。关联数据处理和加密数据处理在时序设计上具有一致性，它们在时序上的步骤划分都能表示为字节展开、数据填充和最终的 P6 排列 3 个部分，其中字节展开和数据填充只和顶层输入相关，只包括简单的数据位展开和异或操作，把这两步组合逻辑合在一起采用一个时钟周期完成。而最终的 P6 排列和上一步的输出结果相关，需要额外进行一个周期的流水线分割。最终数据的处理从输入到输出实现了两个时钟周期的流水线式设计。解密的时序设计和前面两个模块存在一些不同，它由字节展开、密文解密、状态恢复和数据填充 4 个部分组成。结合 FPGA 对数据位操作的优越性，可以将后面 3 个部分直接看作一个整体进行一个时钟周期的数据处理，为字节展开单独分配一个时钟，总体上解密部分也能在两个时钟周期内完成。

局部处理上，对于同一个模块的输入必须在同一时刻开始数据处理，但在设计过程中由于模块之间数据传输，往往会有很多信号不是在同一个时刻到达的，需要对其进行打拍，以达到时序对齐，否则会时序混乱，数据处理错误。以解密模块为例，如图 7 所示，输入信号有 3 个，分别是输入密文参数（ct_adlen）、输入关联数据（ct_sin）、输入密文（ct）。可以看到它们并不是同一时刻到达，输入密文信号具有最大时延，以此为标准，对输入密文参数信号进行 4 个时钟周期的打拍，对输入关联数据进行 2 个时钟周期的打

表 1　　**S 盒优化的查找表**

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	10	11	12	13	14	15	16	17	18	19	1a	1b	1c	1d	1e	1f
S(*x*)	4	b	1f	14	1a	15	9	2	1b	5	8	12	1d	3	6	1c	1e	13	7	e	0	d	11	18	10	c	1	19	16	a	f	17

拍，如图 7 所示打拍后的信号已经和输入密文在同一个时钟沿上对齐，这时再进行流水线设计，使输出相对输入信号延迟一个时钟输出。

2　软件设计

为了验证本文软硬件协同设计在基准三性能指标下的高效性，将前面的 IP 核进行封装打包，搭建系统工程 BlockDsign，构建软硬件协同工作的总体框架，如图 8 所示。其中，在系统配置上将 PL 端工作频率统一配置成 100 MHz，PS 端的 DDR 的读写频率和 PS 端的 CPU 频率分别统一设置为 666.67 MHz 和 533.33 MHz，在 ARM 核上的纯软件测试采取相同的系统配置，ARM 对比实现的系统工程如图 9 所示。模块设计生成顶层包装，编译产生二进制文件，将比特流导入 ARM 核中。再在 PS 端进行软件部分的程序设计，以实现 PS 端与 PL 端的软硬件协同工作。

在 NIST 的基准三里面已经给出了算法的测试文件，该文件主要是针对软件实现情况的测试，还需要根据软硬件协同设计的实际情况对测试文件进行修改以实现 PS 端测试文件与 PL 端硬件 IP 核的协同工作。主要包括两个部分，一是添加 PS 端与 PL 端的发送与接收接口；二是 AXI_Lite 协议规定在其总线上进行数据传输的数据位宽只能是 32 位，对于不符合协议规范的数据类型进行类型转换，以使其能够在 AXI4_Lite 总线上传输。

根据 XILINX 官方手册[14]，AXI4_Lite 总线是基于内存地址映射实现数据传输的，调用 XILINX 官方定义的数据输出函数 Xil_In32 与数据输入函数 Xil_Out32，对函数传入地址参数与待传数据即可实现 PS 端与 PL 端的数据交互。在 AXI4_Lite 的总线协议上定义了一个基地址——BASEADDR，用于指定数据从哪个内存地址开始进行存放，往后的内存地址在基地址基础上加 4 进行内存映射地址的指定协同工作数据交互流程如图 10 所示。

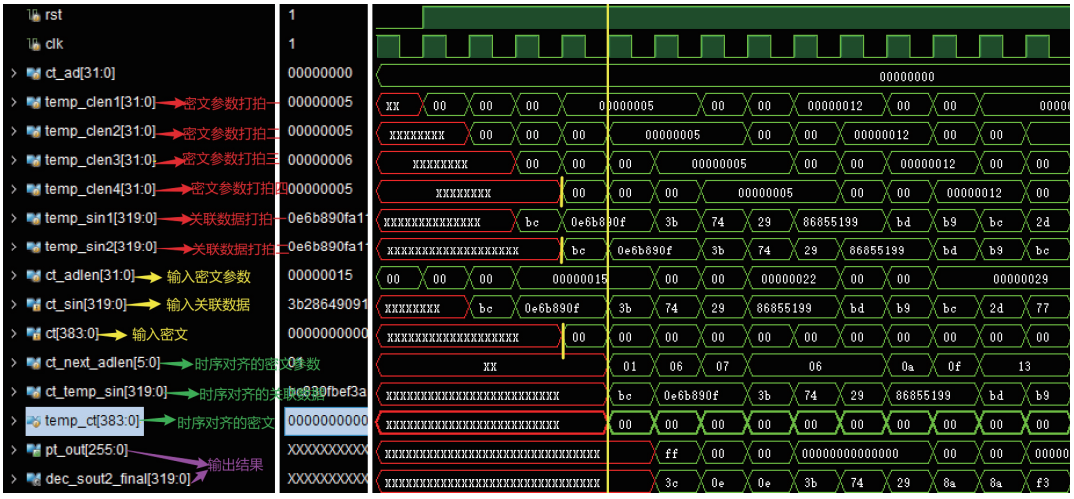


图 7　解密模块时序仿真

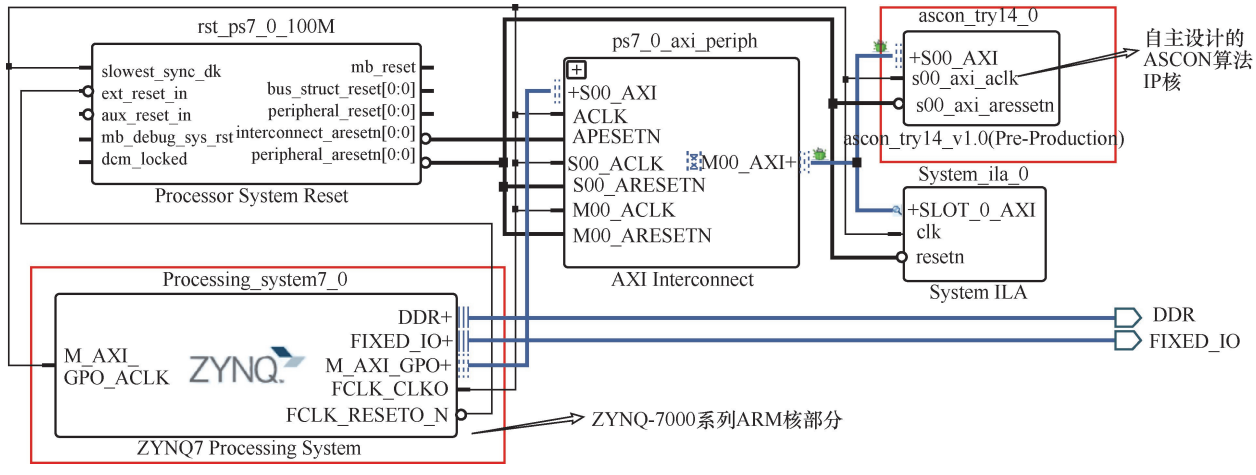


图 8　软硬件协同工作系统设计——Block Design

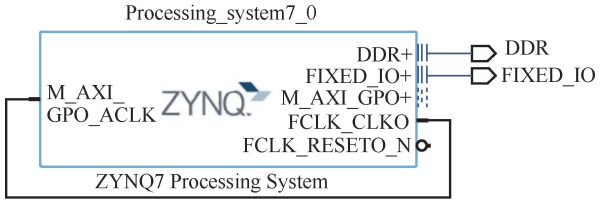


图 9　ARM 核系统设计——Block Design

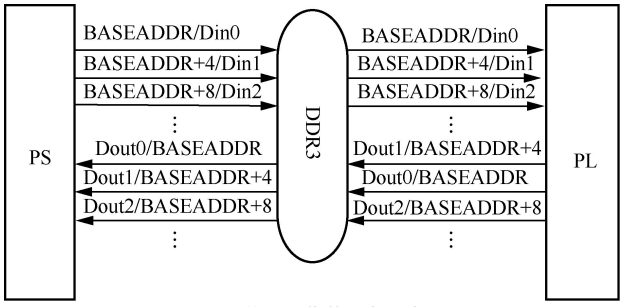


图 10　协同工作数据交互流程

原测试文件中，明文和密钥参数的数据类型为 unsigned char 类型，数据位宽为 8 位，参数采用高位补零的方式将其数据类型定义为 uint32_t，使其数据位宽为 32 位。同理，在接收加解密数据时，原结构体需重新定义使其子成员数据类型为 uint32_t，位宽为 32 位，以完成协同工作的数据接收。

3　性能验证

记录软硬件协同工作下的 PS 端内存消耗和 ASCON 算法加解密时间，通过与 ZYNQ 上 ARM、5 种常用物联网平台的软件实现进行对比，验证设计的高效性。

3.1　程序时间及内存测试方法

根据基准三，测试算法的系统运行时间和内存占用。一次加解密的时间测量可以运用 Xtime_GetTime 函数，函数模型为

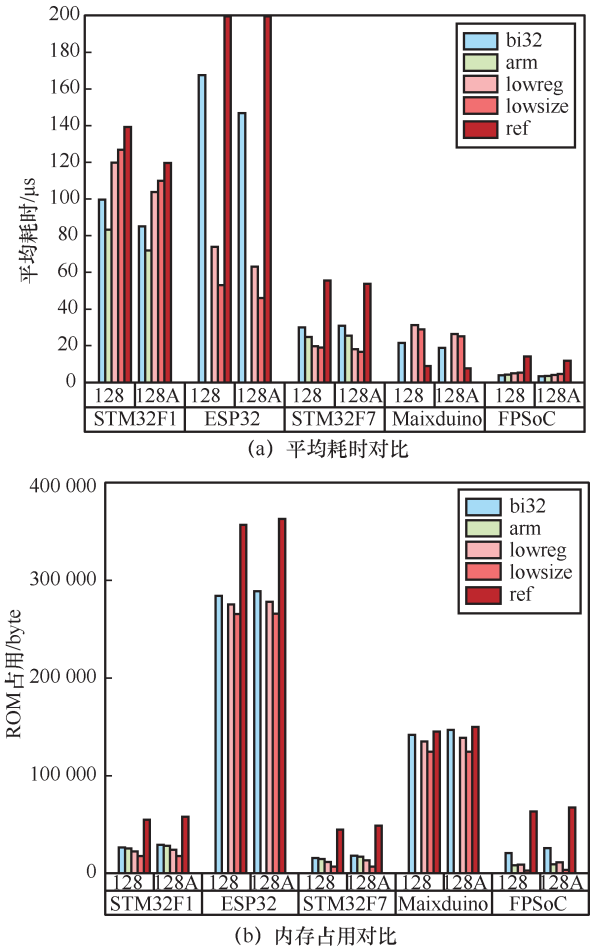
$$T_{\text{used}}=\frac{(T_{\text{end}}-T_{\text{cur}})\times 1\,000\,000}{\text{COUNTS\_PER\_SECOND}} \tag{1}$$

其中，T_{end}、T_{cur} 分别表示待测程序段的结束时间和起始计数值，两者之差即为待测程序的定时器计数的差值，COUNTS_PER_SECOND 表示单位时间的基准计数值。

为得到 ASCON 的 1 089 次加解密的总运行时间，需要对每次加解密时间进行累加输出。采用指针数组的方法，将每次加解密时间的变量指针存入数组，对数组元素累加，即可得到总加解密时间。

算法内存占用主要考虑 ASCON 本身，测试向量文件不算在内。采用的方法是将一个空值 Main.c 放入 ASCON 工程里面，用此时的内存大小减去空值 Main.c 单独占的内存就等于当前 ASCON 算法占用的内存。

ASCON 的多平台软件实现横向对比如图 11 所示；软硬件协同设计与 ARM 核软件实现纵向对比如图 12 所示。



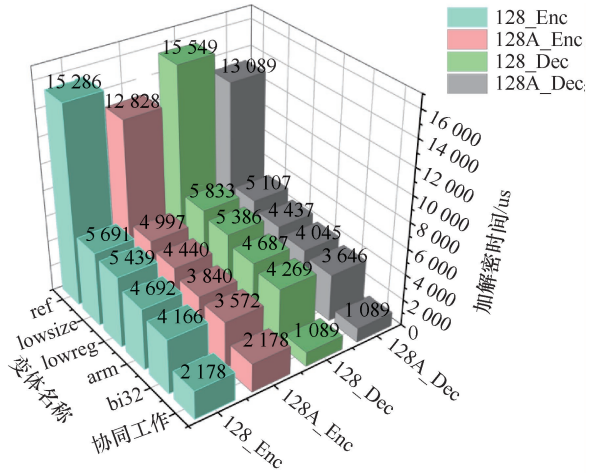
(a) 平均耗时对比

(b) 内存占用对比

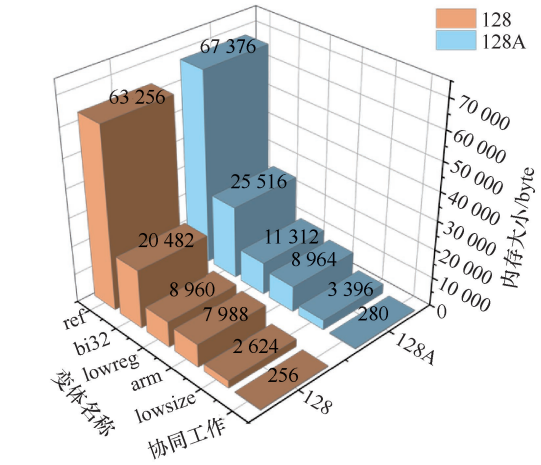
图 11　ASCON 的多平台软件实现横向对比

3.2　性能分析

在软硬件协同工作下，将加解密输出结果和基准已知答案文本进行对比，加解密正确，成功实现 ASCON 软硬件协同工作。在横向对比中，ESP32 在 5 种设备中表现最差，内存占用大、运行时间较长，不适合作为物联网节点。本文所采用的 FPSoC 的 ARM 核测试在运行速度上表现最好，是其他平台实现效果的 4 倍以上；内存管理上 STM32F7 表现最优异，FPSoC 平台表现次之。综合考虑，STM32F7 和 STM32F1 以及本文的 FPSoC 上都有不错的表现，在软件实现的情况下，FPSoC 的 ARM 核实现最优。



(a) 加解密耗时对比



(b) 占用内存对比

图 12　软硬件协同设计与 ARM 核软件实现纵向对比

在软硬件协同设计与 ARM 核的纵向对比中，如图 12 所示，ASCON_128 的软硬件协同工作的实现速度是 ARM 核参考实现（ref）的 9.44 倍，是最快优化实现（bi32）的 2.58 倍；内存上相对于 ARM 核参考实现减少了 99%，比内存大小优化实现（bi32_lowsize）则减小了 90%。在 ASCON_128A 方面，软硬件协同工作的速度是其 ARM 核参考实现速度的 7.93 倍，是其最快优化实现的 2.21 倍；内存上相对于 ARM 核参考实现减少 99%，比内存大小优化实现减少了 92%。综上，本文所提的对 ASCON 的软硬件协同设计在算法运行速度上至少是其在常用物联网设备参考实现的 7.93 倍，内存上至少降低 90%。

ASCON_128 和 ASCON_128A 的在软硬件协同实现上具有相同的实现速度，原因是从 PS 端传到 PL 端的参数数量一致，且在 PL 端进行完一次加密解密所需的时钟周期一致，故两者在速度上一样。内存上由于将整个算法硬件化，在 PS 端只提供数据输入，故内存大大减小，不过与之对应的是 FPGA 资源的消耗，ASCON_128 和 ASCON_128A 的 FPGA 资源消耗分别见表 2、表 3，ASCON_128 消耗了 39 224 片查找表（Slice LUTs），占总查找表的 73.73%；寄存器片（Slice Register）的 7 446 片，占总额的 7%；其他资源消耗有 F7 复用器占用 3.17%，F8 复用器占用 0.5%。ASCON_128A 消耗了查找表的 55.40%共 29 471 片，寄存器片占用 6.16%共 6 555 片，其他资源消耗有 0.76%的 F7 复用器，0.2%的 F8 复用器。

从实验结果来看，ASCON 两种变体的软硬件协同工作的运行速度都比纯软件的无论是参考实现还是优化实现都快了很多，软件部分对设备内存要求也大大降低，这在物联网里对时延、速度、内存要求比较高的环境下具有很大意义。

表 2　ASCON_128 的 FPGA 资源消耗

资源类型	使用量/片	总量/片	使用率
Slice LUTs	39 224	53 299	73.73%
Slice Register	7 446	106 400	7%
F7 复用器	844	26 600	3.17%
F8 复用器	67	13 300	0.5%

表 3　ASCON_128A 的 FPGA 资源消耗

资源类型	使用量/片	总量/片	使用率
Slice LUTs	29 471	53 200	55.40%
Slice Register	6 555	106 400	6.16%
F7 复用器	202	26 600	0.76%
F8 复用器	27	13 300	0.20%

4　结束语

ASCON 算法是一种轻量级认证加密算法，本文以 FPSoC 为研究平台，实现了 ASCON 的软硬件协同工作，并和纯软件实现方式进行对比，结果表明以硬件 IP 为核心的软硬件协同设计具有非常优良的运算时间和内存效率。在当前物联网蓬勃发展的背景下，本文所述的软硬件协同设计方法在物联网传输层的网关，乃至感知层终端中都具有重要的价值。在具体应用中，可以将 ASCON 算法的硬件 IP 核嵌入 FPSoC 的 PL 部分，将数据接口引出到 PS 端，实现向下基于 ASCON 的数据接收与解密和向上的异构网络协议转换打包发送[35]，使整个

FPSoC 成为一个高性能、安全性强、易拓展、可在线升级维护的物联网网关；也可以基于本文对 ASCON 算法的硬件 IP 核设计与验证工作，把系统硬件 IP 移植到 ASIC 上，批量生产基于新一代轻量级认证加密算法 ASCON 的物联网安全芯片。

参考文献：

[1] ALFERIDAH D K, JHANJHI N. A review on security and privacy issues and challenges in internet of things[J]. International Journal of Computer Science and Network Security IJCSNS, 2020, 20(4): 263-86.

[2] ALABA F A, OTHMAN M, HASHEM I A T, et al. Internet of things security: a survey[J]. Journal of Network and Computer Applications, 2017, 88: 10-28.

[3] MOUSAVI S K, GHAFFARI A, BESHARAT S, et al. Security of internet of things based on cryptographic algorithms: a survey[J]. Wireless Networks, 2021, 27(2): 1515-1555.

[4] CHAHAL R K, KUMAR N, BATRA S. Trust management in social internet of things: a taxonomy, open issues, and challenges[J]. Computer Communications, 2020, 150: 13-46.

[5] DIRO A, REDA H, CHILAMKURTI N, et al. Lightweight authenticated-encryption scheme for internet of things based on publish-subscribe communication[J]. IEEE Access, 8: 60539-60551.

[6] HUANG W, LIAO Y J, ZHOU S J, et al. An efficient deniable authenticated encryption scheme for privacy protection[J]. IEEE Access, 2019(7): 43453-43461.

[7] MARTÍNEZ-RODRÍGUEZ M C, SAURO DEL VALLE S, BROX P, et al. Hardware implementation of authenticated ciphers for embedded systems[J]. IEEE Latin America Transactions, 2020, 18(9): 1581-1591.

[8] DOBRAUNIG C, EICHLSEDER M, MENDEL F, et al. Ascon v1.2: lightweight authenticated encryption and hashing[J]. Journal of Cryptology, 2021, 34(3): 1-42.

[9] FOTOVVAT A, RAHMAN G M E, VEDAEI S S, et al. Comparative performance analysis of lightweight cryptography algorithms for IoT sensor nodes[J]. IEEE Internet of Things Journal, 2021, 8(10): 8279-8290.

[10] SONMEZ TURAN M, MCKAY K, CHANG D, et al. Status report on the second round of the NIST lightweight cryptography standardization process[R]. National Institute of Standards and Technology, 2021.

[11] DEGABRIELE J P, JANSON C, STRUCK P. Sponges resist leakage: the case of authenticated encryption[M]//Lecture Notesin Computer Science. Cham: Springer International Publishing, 2019: 209-240.

[12] DALMASSO L, BRUGUIER F, BENOIT P, et al. Evaluation of SPN-based lightweight crypto-ciphers[J]. IEEE Access, 2019(7): 10559-10567.

[13] MOLANES R F, COSTAS L, RODRÍGUEZ-ANDINA J J, et al. Comparative analysis of processor-FPGA communication performance in low-cost FPSoCs[J]. IEEE Transactions on Industrial Informatics, 2021, 17(6): 3826-3835.

[14] XILINX I. Xilinx Zynq-7000 SoC Technical Reference Manual[EB]. 2021.

[15] TEMPELMEIER M, DE SANTIS F, SIGL G, et al. The CAESAR-API in the real world—towards a fair evaluation of hardware CAESAR candidates[C]//Proceedings of 2018 IEEE International Symposium on Hardware Oriented Security and Trust. Piscataway: IEEE Press, 2018: 73-80.

[16] 潘新祥, 胡习霜, 韩立宏. 软硬件协同设计分析[J]. 指挥控制与仿真, 2008, 30(3): 117-119.
PAN X X, HU X S, HAN L H. Analysis on designation in cooperation with hardware-software[J]. Command Control & Simulation, 2008, 30(3): 117-119.

[17] COWART R, COE D, KULICK J, et al. An implementation and experimental evaluation of hardware accelerated ciphers in all-programmable SoCs[C]//Proceedings of ACM SE '17: Proceedings of the SouthEast Conference. New York：ACM Press, 2017: 34-41.

[18] FERNANDEZ MOLANES R, AMARASINGHE K, RODRIGUEZ-ANDINA J, et al. Deep learning and reconfigurable platforms in the internet of things: challenges and opportunities in algorithms and hardware[J]. IEEE Industrial Electronics Magazine, 2018, 12(2): 36-49

[19] ZHAI X J, ALI A A S, AMIRA A, et al. MLP neural network based gas classification system on zynq SoC[J]. IEEE Access, 2016(4): 8138-8146.

[20] SUMARUDIN A, ADIONO T, PUTRA W P. Flexible and reconfigurable system on chip for wireless sensor network[C]//Proceedings of 2014 International Conference on Information Technology Systems and Innovation (ICITSI). Piscataway: IEEE Press, 2014: 230-234.

[21] RUCKEBUSCH P, GIANNOULIS S, GARLISI D, et al. WiSHFUL: enabling coordination solutions for managing heterogeneous wireless networks[J]. IEEE Communications Magazine, 2017, 55(9): 118-125.

[22] AITSIALIA, FARHAT A, MOHAMAD S, et al. Embedded platform for gas applications using hardware/software co-design and RFID[J]. IEEE Sensors Journal, 2018, 18(11): 4633-4642.

[23] NIST. NIST LWC software performance benchmarks on microcontrollers[EB]. 2020.

[24] 李玉波. 基于 ARM 体系看嵌入式处理器的发展[J]. 电子技术与软件工程, 2016(11): 213.
LI Y B. Development of embedded processor based on ARM system[J]. Electronic Technology & Software Engineering, 2016(11): 213.

[25] CARDOSO DOS SANTOS L, GROßSCHäDL J. An evaluation of the multi-platform efficiency of lightweight cryptographic permutations[C]//Proceedings of the International Conference on Information Technology and Communications Security. [S.L.:s.n], 2022: 70-85.

[26] 周朕, 何德彪, 罗敏, 等. 紧凑的 Aigis-sig 数字签名方案软硬件协同实现方法[J]. 网络与信息安全学报, 2021, 7(2): 64-76.
ZHOU I, HE D B, LUO M, et al. Compact Aigis-sig digital signature scheme based on software and hardware collaboration[J]. Journal of Network and Information Security, 2017, 7(2):64-76.

[27] 钟震宇. 基于 Python 硬件描述的 AXI4 总线接口设计与实现[D]. 广州: 华南理工大学, 2020.
ZHONG Z Y. Design and implementation of AXI4 bus interface based on python hardware description[D]. Guangzhou: South China University of Technology, 2020.

[28] 许杰, 张子恒, 王新宇, 等. 一种基于 Zynq 的 CNN 加速器设计与实现[J]. 计算机技术与发展, 2021, 31(11): 108-113, 121.
XU J, ZHANG Z H, WANG X Y, et al. Design and implementation of CNN accelerator based on zynq[J]. Computer Technology and Development, 2021, 31(11): 108-113, 121.

[29] 刘 祥. 基于加密算法的软硬件协同设计与实现及云安全存储研究[D]. 广州: 广东工业大学, 2020.
LIU X. Software-hardware collaborative design and implementation based on encryption algorithm and cloud secure storage[D]. Guangzhou: Guangdong University of Technology, 2020.

[30] 张盛仕, 胡湘宏, 熊晓明. 基于国密算法 SM2 软硬件协同系统的

FPGA 架构[J]. 单片机与嵌入式系统应用, 2019, 19(7): 15-19.
ZHANG S S, HU X H, XIONG X M. FPGA architecture of software and hard ware co-design based on national secret algorithm SM2[J]. Microcontrollers＆ Embedded Systems, 2019, 19(7): 15-19.

[31] KAUR J, MOZAFFARIKERMANI M, AZARDERAKHSH R. Hardware constructions for error detection in lightweight authenticated cipher ASCON benchmarked on FPGA[J]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2022, 69(4): 2276-2280.

[32] GROSS H, WENGER E, DOBRAUNIG C, et al. Ascon hardware implementations and side-channel evaluation[J]. Microprocessors and Microsystems, 2017, 52: 470-479.

[33] 刘玉宣. 基于 FPGA 的高性能椭圆曲线密码加速技术研究[D]. 合肥: 合肥工业大学, 2021.
LIU Y X. Research on FPGA-based high-performance elliptic curve cryptography acceleration technology[D]. Hefei: Hefei University of Technology, 2021.

[34] 方轶, 丛林虎, 邓建球, 等. 基于FPGA的SM3算法快速实现方案[J]. 计算机应用与软件, 2020, 37(6): 259-262.
FANG Y, CONG L H, DENG J Q, et al. Fast implementation of Sm3 algorithm based on FPGA[J]. Computer Applications and Software, 2020, 37(6): 259-262.

[35] 史冰清. 高安全性的物联网网关设计与实现[D]. 成都: 电子科技大学, 2018.
SHI B Q. Design and implementation of IoT gateway for high security[D]. Chengdu: University of Electronic Science and Technology of China, 2018.

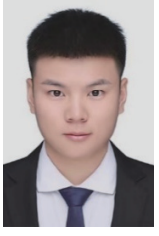
[作者简介]

汪静（1998− ），男，云南大学信息学院硕士生，主要研究方向为嵌入式开发与物联网安全。

何乐生（1977− ），男，博士，云南大学信息学院副教授，主要研究方向为嵌入式系统及物联网应用、微弱信号采集和处理及其在生物电信号和射电天文信号处理等。

李忠红（1995− ），男，云南大学信息学院硕士生，主要研究方向为能源物联网、嵌入式开发。

李路迟（1998− ），女，云南大学信息学院硕士生，主要研究方向为区块链共识机制与物联网安全。



杨航（1997− ），男，云南大学信息学院硕士生，主要研究方向为嵌入式开发与射电天文无线电环境监测。